**Цели и методы рефакторинга**

*Refactoring* – это процесс изменения структуры программы, в не изменяющий её поведения, но облегчающий понимание её работы и модификацию программы. Рефакторинг делает код более удобным, понятным и легким в обслуживании.

Цель рефакторинга: повышение качество ПО с точки зрения сопровождаемости.

Отличие рефакторинга от оптимизации производительности: рефакторинг и оптимизация не меняют поведение системы, но оптимизация может привести к затруднению понимания кода.

Отличие рефакторинга от реинжиниринга:

Реинжиниринг осуществляется для расширения функциональности ПО. Обычно рефакторинг выполняется перед реинжинирингом.

Причины использования рефакторинга:

1. Необходимость добавить новую функциональность, которая не укладывается в принятое архитектурное решение.
2. Необходимость исправить ошибку, причина возникновения которой сразу непонятна.
3. Преодоление трудностей, связанных со сложной логикой ПО и длительным процессом разработки.

Рекомендации пере выполнением рефакторинга:

1. Сохранить копию исходного кода.
2. Желательно покрыть изменяемую часть кода модульными тестами, чтобы не сломать что работало до нас.

Группы приемов рефакторинга:

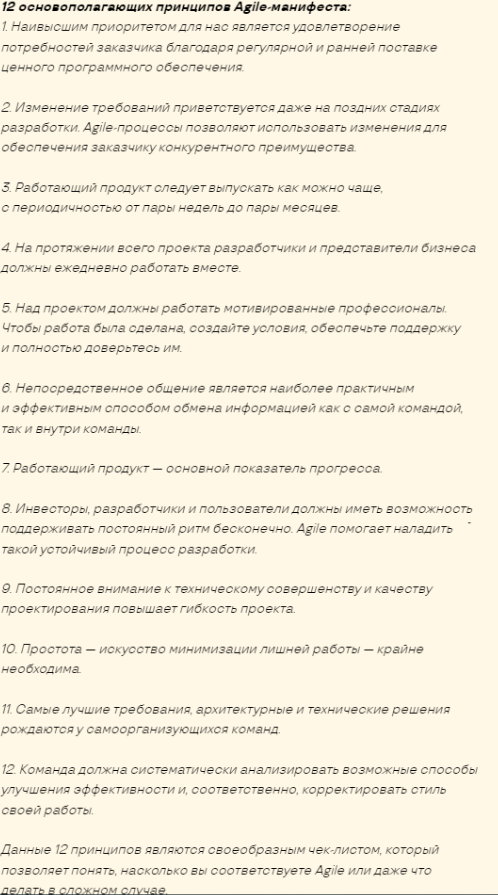
1. Составление методов. Используется для понижения сложности внутри метода, избавление от дублирования кода и облегчение последующей работы с кодом.
2. Перемещение функций между объектами. Используется для перемещения функциональности из одних классов в другие, создание новых классов и скрытие деталей реализации из публичного доступа.
3. Организация данных. Используется для облегчения работы с данными и уменьшения связанности между классами.
4. Упрощение условных выражений.
5. Упрощение вызовов методов.
6. Решение задач обобщения.

Рефакторинг является частью жизненного цикла ПО, который разрабатывается по методологии Agile.

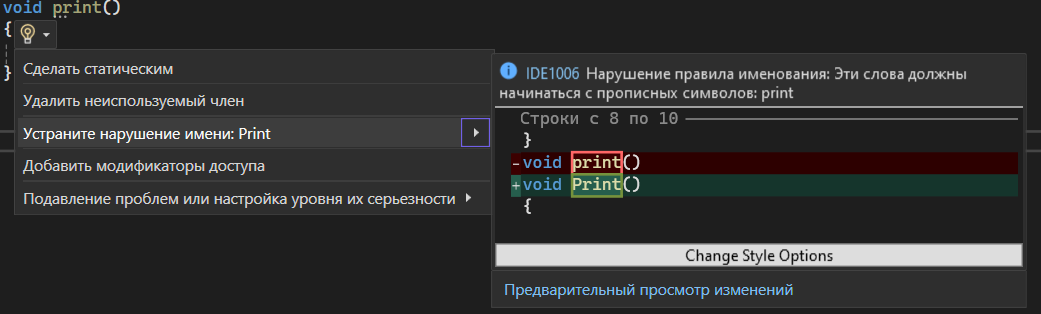


Для того, чтобы процесс рефакторинга не ухудшал способности ПО, код надо покрывать модульными тестами

Принципы Agile:



В Agile обычно включаются возможности рефакторинга



Комбинации:

Ctrl+R+R – переименовка

Ctrl+R+M

Ctrl+R+E – инкапсуляция поля. На основе поля создается свойство

Ctrl+K+E – метелка

Alt+Enter – вызывает окно рефакторинга

ReSharper – это плагин для VS

Документирование кода:

Code Style:

PascalCase – типы данных и функции типы данных

camelCase – поля, параметры методов, переменные

\_age – закрытое поле

LoginTextBox / loginTextBox – поля ввода

LoginTextBox\_TextChanged <= TextChanged – метод, поэтому с большой

или

LoginTextBoxOnTextChanged

Пример:

BooksForm

RegistrationWindows

Везде где хранятся данные первое слово существительное

OpenButton

CancelButton / OkButton

Аббревиатура капслоком если меньше трех символов

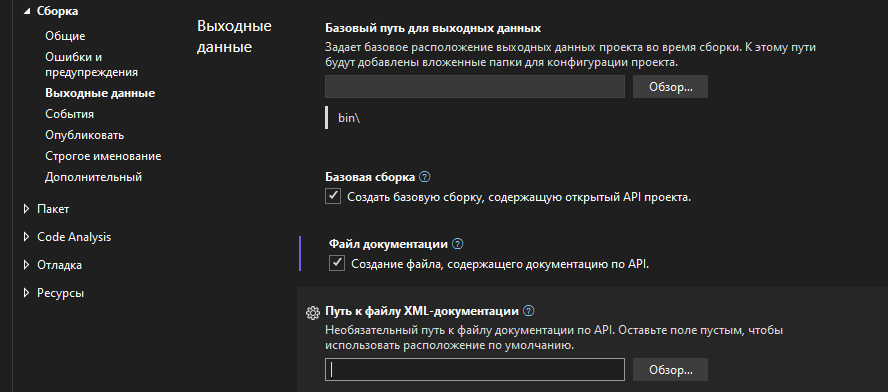
System.IO;

HttpListener

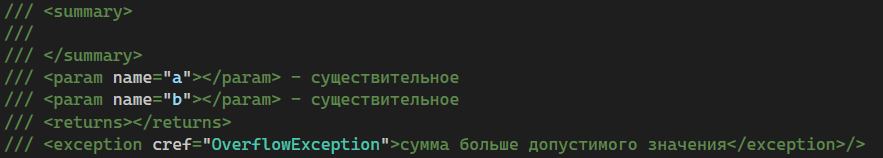
Api

XML-документация



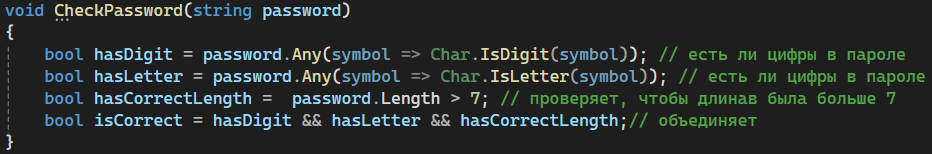


Содержимое тегов



Примеры рефакторингов:

1. Извлечение переменной. Помогает упростить сложное выражение



1. Замена сложного алгоритма. Замена алгоритма, который проще текущего

Вместо цикла использовать формулу:

*sum = (first+last)\*n/2;*

1. Замена кода ошибки исключений

Возвращать код ошибки устаревшая практика

